Glossary of Java and Related Terms

100% Pure Java(TM);

A Sun Microsystems initiative to guide developers in writing, certifying, and marketing applications written entirely in the Java(TM) programming language.

A

Abstract Window Toolkit (AWT)

A collection of graphical user interface (GUI) components that were implemented using native-platform versions of the components. These components provide that subset of functionality which is common to all native platforms. Largely supplanted by the Project Swing component set. See also *Swing Set*.

abstract

A Java(TM) programming language keyword used in a class definition to specify that a class is not to be instantiated, but rather inherited by other classes. An abstract class can have abstract methods that are not implemented in the abstract class, but in subclasses.

abstract class

A class that contains one or more *abstract methods*, and therefore can never be instantiated. Abstract classes are defined so that other classes can extend them and make them concrete by implementing the abstract methods.

abstract method

A method that has no implementation.

access control

The methods by which interactions with resources are limited to collections of users or programs for the purpose of enforcing integrity, confidentiality, or availability constraints.

ACID

The acronym for the four properties guaranteed by transactions: atomicity, consistency, isolation, and durability.

activation

The process of transferring an enterprise bean from secondary storage to memory. (See passivation.)

actual parameter list

The arguments specified in a particular method call. See also *formal parameter list*.

alpha value

A value that indicates the opacity of a pixel.

API

Application Programming Interface. The specification of how a programmer writing an application accesses the behavior and state of classes and objects.

applet

A component that typically executes in a Web browser, but can execute in a variety of other applications or devices that support the applet programming model.

Applet container

A container that includes support for the applet programming model.

appliances

Networked devices such as printers, Java(TM) technology-enabled terminals, and clients, that are managed using applications built using the Java Management API (JMAPI).

Application Assembler

A person that combines components and modules into larger deployable application units.

application client

A first-tier client program that executes in its own Java Virtual Machine.

application client container

A container that supports application clients and provides a federated view of the J2EE platform APIs.

application client module

A software unit that consists of one or more classes and an application client deployment descriptor.

Application Component Provider

A vendor that provides the Java classes that implement components' methods, JSP page definitions, and any required deployment descriptors.

APM

Application Programming Model. A programming model that defines how to use and combine the features of the J2EE platform to create solutions for common application domains in the enterprise.

argument

A data item specified in a method call. An argument can be a literal value, a variable, or an expression.

array

A collection of data items, all of the same type, in which each item's position is uniquely designated by an integer.

ASCII

American Standard Code for Information Interchange. A standard assignment of 7-bit numeric codes to characters. See also *Unicode*.

atomic

Refers to an operation that is never interrupted or left in an incomplete state under any circumstance.

authentication

The process by which an entity proves to another entity that it is acting on behalf of a specific identity. The J2EE platform requires three types of authentication: basic, form-based, and mutual.

authorization

See access control.

authorization constraint

A set of security roles that are allowed access to the resources in a Web resource collection.

B

basic authentication

When a Web server authenticates an entity with a user name and password obtained using the Web client's built-in authentication mechanism.

Bean

A reusable software component. Beans can be combined to create an application.

bean-managed persistence

When the transfer of data between an entity bean instance's variables and the underlying resource manager is managed by the entity bean.

bean-managed transaction

When an enterprise bean defines the boundaries of the transaction.

bean-managed transaction

When an enterprise bean defines the boundaries of the transaction.

binary operator

An operator that has two arguments.

bit

The smallest unit of information in a computer, with a value of either 0 or 1.

bitwise operator

An operator that manipulates the bits of one or more of its operands individually and in parallel. Examples include the binary logical operators (&, |, ^), the binary shift operators (<<, >>, >>>) and the unary one's complement operator (~).

block

In the Java(TM) programming language, any code between matching braces. Example: `{ x = 1; }`.

boolean

Refers to an expression or variable that can have only a true or false value. The Java(TM) programming language provides the boolean type and the literal values true and false.

bounding box

The smallest rectangle that encloses a geometric shape. For a Raster object, the smallest rectangle that completely encloses all the defined pixels.

break

A Java(TM) programming language keyword used to resume program execution at the statement immediately following the current statement. If followed by a label, the program resumes execution at the labeled statement.

business logic

The code that implements the functionality of an application. In the Enterprise JavaBeans model, this logic is implemented by the methods of an enterprise bean.

business method

A method of an enterprise bean that implements the business logic or rules of an application.

byte

A sequence of eight bits. The Java(TM) programming language provides a corresponding `byte` type.

bytecode

Machine-independent code generated by the Java(TM) compiler and executed by the Java interpreter.

C

callback methods

Methods in a component called by the container to notify the component of important events in its life cycle.

caller

Same as caller principal.

caller principal

The principal that identifies the invoker of an enterprise bean method.

case

A Java(TM) programming language keyword that defines a group of statements to begin executing if a value specified matches the value defined by a preceding "switch" keyword.

casting

Explicit conversion from one data type to another.

catch

A Java(TM) programming language keyword used to declare a block of statements to be executed in the event that a Java exception, or run time error, occurs in a preceding "try" block.

char

A Java(TM) programming language keyword used to declare a variable of type character.

class

In the Java(TM) programming language, a type that defines the implementation of a particular kind of object. A class definition defines instance and class variables and methods, as well as specifying the interfaces the class implements and the immediate superclass of the class. If the superclass is not explicitly specified, the superclass will implicitly be `Object`.

class method

A method that is invoked without reference to a particular object. Class methods affect the class as a whole, not a particular instance of the class. Also called a *static method*. See also *instance method*.

classpath

A classpath is an environmental variable which tells the Java(TM) virtual machine* and Java technology-based applications (for example, the tools located in the JDK(TM) 1.1.X\bin directory) where to find the class libraries, including user-defined class libraries.

class variable

A data item associated with a particular class as a whole--not with particular instances of the class. Class variables are defined in class definitions. Also called a *static field*. See also *instance variable*.

client

In the client/server model of communcations, the client is a process that remotely accesses resources of a compute server, such as compute power and large memory capacity.

codebase

Works together with the `code` attribute in the `<APPLET>` tag to give a complete specification of where to find the main applet class file: code specifies the name of the file, and codebase specifies the URL of the directory containing the file.

comment

In a program, explanatory text that is ignored by the compiler. In programs written in the Java(TM) programming language, comments are delimited using `//` or `/*...*/`.

commit

The point in a transaction when all updates to any resources involved in the transaction are made permanent.

compilation unit

The smallest unit of source code that can be compiled. In the current implementation of the Java(TM) platform, the compilation unit is a file.

compiler

A program to translate source code into code to be executed by a computer. The Java(TM) compiler translates source code written in the Java programming language into bytecode for the Java virtual machine*. See also *interpreter*.

component

An application-level software unit supported by a container. Components are configurable at deployment time. The J2EE platform defines four types of components: enterprise beans, Web components, applets, and application clients.

component contract

The contract between a component and its container. The contract includes: life cycle management of the component, a context interface that the instance uses to obtain various information and services from its container, and a list of services that every container must provide for its components.

component environment

A set of requirements defined by the Application Component Provider required to be available to a J2EE component in its naming environment. The environment entries are declaratively specified in

the component's deployment descriptor. Each component names and accesses its environment configuration values using the java:comp/env JNDI context. These values can be objects a component is dependent on, such as a JDBC DataSource or a simple value such as a tax rate.

compositing

The process of superimposing one image on another to create a single image.

connection

See resource manager connection.

connection factory

See resource manager connection factory.

connector

A standard extension mechanism for containers to provide connectivity to EISs. A connector is specific to an EIS and consists of a resource adapter and application development tools for EIS connectivity. The resource adapter is plugged in to a container through its support for system level contracts defined in the connector architecture.

connector architecture

An architecture for integration of J2EE servers with EISs. There are two parts to this architecture: a EIS vendor-provided resource adapter and a J2EE server that allows this resource adapter to plug in. This architecture defines a set of contracts that a resource adapter has to support to plug in to a J2EE server, for example, transactions, security, resource management.

constructor

A pseudo-method that creates an object. In the Java(TM) programming language, constructors are instance methods with the same name as their class. Constructors are invoked using the `new` keyword.

const

This is a reserved Java(TM) programming language keyword. However, it is not used by current versions of the Java programming language.

container

An entity that provides life cycle management, security, deployment, and runtime services to components. Each type of container (EJB, Web, JSP, servlet, applet, and application client) also provides component-specific services.

container-managed persistence

When transfer of data between an entity bean's variables and the underlying resource manager is managed by the enterprise bean's container.

container-managed transaction

When an EJB container defines the boundaries of a transaction. An entity bean must use container-managed transactions.

context attribute

An object bound into the context associated with a servlet.

continue

A Java(TM) programming language keyword used to resume program execution at the end of the current loop. If followed by a label, "continue" resumes execution where the label occurs.

conversational state

The field values of a session bean plus the transitive closure of the objects reachable from the bean's fields. The transitive closure of a bean is defined in terms of the serialization protocol for the Java programming language, that is, the fields that would be stored by serializing the bean instance.

CORBA

Common Object Request Broker Architecture. A language independent, distributed object model specified by the Object Management Group (OMG).

core class

A public class (or interface) that is a standard member of the Java(TM) Platform. The intent is that the core classes for the Java platform, at minimum, are available on all operating systems where the Java platform runs. A program written entirely in the Java programming language relies only on core classes, meaning it can run anywhere. See also, *100% Pure Java(TM)*.

Core Packages

The required set of APIs in a Java platform edition which must be supported in any and all compatible implementations.

create method

A method defined in the home interface and invoked by a client to create an enterprise bean.

credentials

The information describing the security attributes of a principal. Credentials can be acquired only through authentication or delegation.

critical section

A segment of code in which a thread uses resources (such as certain instance variables) that can be used by other threads, but that must not be used by them at the same time.

CTS

Compatibility Test Suite. A suite of compatibility tests for verifying that a J2EE product complies with the J2EE platform specification.

D

declaration

A statement that establishes an identifier and associates attributes with it, without necessarily reserving its storage (for data) or providing the implementation (for methods). See also *definition*.

default

A Java(TM) programming language keyword optionally used after all "case" conditions in a "switch" statement. If all "case" conditions are not matched by the value of the "switch" variable, the "default" keyword will be executed.

definition

A declaration that reserves storage (for data) or provides implementation (for methods). See also

declaration.

delegation

An act whereby one principal authorizes another principal to use its identity or privileges with some restrictions.

Deployer

A person who installs modules and J2EE applications into an operational environment.

deployment

The process whereby software is installed into an operational environment.

deployment descriptor

An XML file provided with each module and application that describes how they should be deployed. The deployment descriptor directs a deployment tool to deploy a module or application with specific container options and describes specific configuration requirements that a deployer must resolve.

deprecation

Refers to a class, interface, constructor, method or field that is no longer recommended, and may cease to exist in a future version.

derived from

Class X is "derived from" class Y if class X *extends* class Y. See also *subclass*, *superclass*.

distributed

Running in more than one address space.

distributed application

An application made up of distinct components running in separate runtime environments, usually on different platforms connected via a network. Typical distributed applications are two-tier (client/server), three-tier (client/middleware/server), and n-tier (client/multiple middleware/multiple servers).

do

A Java(TM) programming language keyword used to declare a loop that will iterate a block of statements. The loop`s exit condition can be specified with the "while" keyword.

DOM

Document Object Model. A tree of objects with interfaces for traversing the tree and writing an XML version of it, as defined by the W3C specification.

double

A Java(TM) programming language keyword used to define a variable of type double.

double precision

In the Java(TM) programming language specification, describes a floating point number that holds 64 bits of data. See also *single precision*.

DTD

Document Type Definition. A description of the structure and properties of a class of XML files.

E

EJB container

A container that implements the EJB component contract of the J2EE architecture. This contract specifies a runtime environment for enterprise beans that includes security, concurrency, life cycle management, transaction, deployment, and other services. An EJB container is provided by an EJB or J2EE server.

EJB Container Provider

A vendor that supplies an EJB container.

EJB context

An object that allows an enterprise bean to invoke services provided by the container and to obtain the information about the caller of a client-invoked method.

EJB home object

An object that provides the life cycle operations (create, remove, find) for an enterprise bean. The class for the EJB home object is generated by the container's deployment tools. The EJB home object implements the enterprise bean's home interface. The client references an EJB home object to perform life cycle operations on an EJB object. The client uses JNDI to locate an EJB home object.

EJB .jar file

A JAR archive that contains an EJB module.

EJB module

A software unit that consists of one or more enterprise beans and an EJB deployment descriptor.

EJB object

An object whose class implements the enterprise bean's remote interface. A client never references an enterprise bean instance directly; a client always references an EJB object. The class of an EJB object is generated by the container's deployment tools.

EJB server

Software that provides services to an EJB container. For example, an EJB container typically relies on a transaction manager that is part of the EJB server to perform the two-phase commit across all the participating resource managers. The J2EE architecture assumes that an EJB container is hosted by an EJB server from the same vendor, so does not specify the contract between these two entities. An EJB server may host one or more EJB containers.

EJB Server Provider

A vendor that supplies the EJB server.

EIS resource

An entity that provides EIS-specific functionality to its clients. Examples are: a record or set of records in a database system, a business object in an ERP system, and a transaction program in a transaction processing system.

else

A Java(TM) programming language keyword used to execute a block of statements in the case that

the test condition with the "if" keyword evaluates to false.

EmbeddedJava(TM) Technology

The availability of Sun's Java 2 Platform, Micro Edition technology under a restrictive license agreement that allows a licensee to leverage certain Java technologies to create and deploy a closed-box application that exposes no APIs.

encapsulation

The localization of knowledge within a module. Because objects encapsulate data and implementation, the user of an object can view the object as a black box that provides services. Instance variables and methods can be added, deleted, or changed, but as long as the services provided by the object remain the same, code that uses the object can continue to use it without being rewritten. See also *instance variable*, *instance method*.

enterprise bean

A component that implements a business task or business entity; either an entity beans or a session bean.

Enterprise Information System (EIS)

The applications that comprise an enterprise's existing system for handling company-wide information. These applications provide an information infrastructure for an enterprise. An EIS offers a well defined set of services to its clients. These services are exposed to clients as local and/or remote interfaces. Examples of EISs include: an ERP system, a mainframe transaction processing system, and a legacy database system.

Enterprise Bean Provider

An application programmer who produces enterprise bean classes, remote and home interfaces, and deployment descriptor files, and packages them in an EJB .jar file.

Enterprise JavaBeans(TM) (EJB)

A component architecture for the development and deployment of object-oriented, distributed, enterprise-level applications. Applications written using the Enterprise JavaBeans architecture are scalable, transactional, and multi-user and secure.

entity bean

An enterprise bean that represents persistent data maintained in a database. An entity bean can manage its own persistence or it can delegate this function to its container. An entity bean is identified by a primary key. If the container in which an entity bean is hosted crashes, the entity bean, its primary key, and any remote references survive the crash.

exception

An event during program execution that prevents the program from continuing normally; generally, an error. The Java(TM) programming language supports exceptions with the try, catch, and throw keywords. See also *exception handler*.

exception handler

A block of code that reacts to a specific type of *exception*. If the exception is for an error that the program can recover from, the program can resume executing after the exception handler has executed.

executable content

An application that runs from within an HTML file. See also *applet*.

extends

Class X extends class Y to add functionality, either by adding fields or methods to class Y, or by overriding methods of class Y. An interface extends another interface by adding methods. Class X is said to be a subclass of class Y. See also *derived from*.

F

finder method

A method defined in the home interface and invoked by a client to locate an entity bean.

FCS

First Customer Ship. The day in which a product is released/shipped to the customer.

field

A data member of a class. Unless specified otherwise, a field is not static.

final

A Java(TM) programming language keyword. You define an entity once and cannot change it or derive from it later. More specifically: a final class cannot be subclassed, a final method cannot be overridden and a final variable cannot change from its initialized value.

finally

A Java(TM) programming language keyword that executes a block of statements regardless of whether a Java Exception, or run time error, occurred in a block defined previously by the "try" keyword.

float

A Java(TM) programming language keyword used to define a floating point number variable.

for

A Java(TM) programming language keyword used to declare a loop that reiterates statements. The programmer can specify the statements to be executed, exit conditions, and initialization variables for the loop.

form-based authentication

When a Web container provides an application-specific form for logging in.

FTP

The basic Internet File Transfer Protocol. FTP, which is based on TCP/IP, enables the fetching and storing of files between hosts on the Internet. See also *TCP/IP*.

formal parameter list

The parameters specified in the definition of a particular method. See also *actual parameter list*.

G

garbage collection

The automatic detection and freeing of memory that is no longer in use. The Java(TM) runtime system performs garbage collection so that programmers never explicitly free objects.

goto

This is a reserved Java(TM) programming language keyword. However, it is not used by current versions of the Java programming language.

group

A collection of principals within a given security policy domain.

GUI

Graphical User Interface. Refers to the techniques involved in using graphics, along with a keyboard and a mouse, to provide an easy-to-use interface to some program.

H

handle

An object that identifies an enterprise bean. A client may serialize the handle, and then later deserialize it to obtain a reference to the enterprise bean.

hexadecimal

The numbering system that uses 16 as its base. The marks 0-9 and a-f (or equivalently A-F) represent the digits 0 through 15. In programs written in the Java(TM) programming language, hexadecimal numbers must be preceded with 0x. See also *octal*.

hierarchy

A classification of relationships in which each item except the top one (known as the root) is a specialized form of the item above it. Each item can have one or more items below it in the hierarchy. In the Java(TM) class hierarchy, the root is the `Object` class.

home interface

One of two interfaces for an enterprise bean. The home interface defines zero or more methods for creating and removing an enterprise bean. For session beans, the home interface defines create and remove methods, while for entity beans, the home interface defines create, finder, and remove methods.

home handle

An object that can be used to obtain a reference of the home interface. A home handle can be serialized and written to stable storage and deserialized to obtain the reference.

HotJava(TM) Browser

An easily customizable Web browser developed by Sun Microsystems, which is written in the Java(TM) programming language.

HTML

HyperText Markup Language. This is a file format, based on SGML, for hypertext documents on the Internet. It is very simple and allows for the embedding of images, sounds, video streams, form fields and simple text formatting. References to other objects are embedded using URLs.

HTTP

HyperText Transfer Protocol. The Internet protocol, based on TCP/IP, used to fetch hypertext objects from remote hosts. See also *TCP/IP*.

HTTPS

HTTP layered over the SSL protocol.

I

IDL

Interface Definition Language. APIs written in the Java(TM) programming language that provide standards-based interoperability and connectivity with CORBA (Common Object Request Broker Architecture).

identifier

The name of an item in a program written in the Java(TM) programming language.

IIOP

Internet Inter-ORB Protocol. A protocol used for communication between CORBA object request brokers.

if

A Java(TM) programming language keyword used to conduct a conditional test and execute a block of statements if the test evaluates to true.

impersonation

An act whereby one entity assumes the identity and privileges of another entity without restrictions and without any indication visible to the recipients of the impersonator's calls that delegation has taken place. Impersonation is a case of simple delegation.

implements

A Java(TM) programming language keyword optionally included in the class declaration to specify any interfaces that are implemented by the current class.

import

A Java(TM) programming language keyword used at the beginning of a source file that can specify classes or entire packages to be referred to later without including their package names in the reference.

inheritance

The concept of classes automatically containing the variables and methods defined in their *supertypes*. See also *superclass*, *subclass*.

initialization parameter

A parameter that initializes the context associated with a servlet.

instance

An object of a particular class. In programs written in the Java(TM) programming language, an instance of a class is created using the `new` operator followed by the class name.

instance method

Any method that is invoked with respect to an instance of a class. Also called simply a *method*. See also *class method*.

instance variable

Any item of data that is associated with a particular object. Each instance of a class has its own

copy of the instance variables defined in the class. Also called a *field*. See also *class variable*.

instanceof

A two-argument Java(TM) programming language keyword that tests whether the run-time type of its first argument is assignment compatible with its second argument.

int

A Java(TM) programming language keyword used to define a variable of type integer.

interface

A Java(TM) programming language keyword used to define a collection of method definitions and constant values. It can later be implemented by classes that define this interface with the "implements" keyword.

Internet

An enormous network consisting of literally millions of hosts from many organizations and countries around the world. It is physically put together from many smaller networks and data travels by a common set of protocols.

IP

Internet Protocol. The basic protocol of the Internet. It enables the unreliable delivery of individual packets from one host to another. It makes no guarantees about whether or not the packet will be delivered, how long it will take, or if multiple packets will arrive in the order they were sent. Protocols built on top of this add the notions of connection and reliability. See also *TCP/IP*.

interpreter

A module that alternately decodes and executes every statement in some body of code. The Java(TM) interpreter decodes and executes bytecode for the Java virtual machine*****. See also *compiler*, *runtime system*.

ISV

Independent Software Vendor.

J

J2EE application

Any deployable unit of J2EE functionality. This can be a single module or a group of modules packaged into an .ear file with a J2EE application deployment descriptor. J2EE applications are typically engineered to be distributed across multiple computing tiers.

J2EE product

An implementation that conforms to the J2EE platform specification.

J2EE Product Provider

A vendor that supplies a J2EE product.

J2EE server

The runtime portion of a J2EE product. A J2EE server provides Web and/or EJB containers.

JAE

Java(TM) Application Environment. The source code release of the Java Development Kit

(JDK(TM)) software.

JAR Files (.jar)

Java ARchive. A file format used for aggregating many files into one.

JAR file format

JAR (Java Archive) is a platform-independent file format that aggregates many files into one. Multiple applets written in the Java(TM) programming language, and their requisite components (.class files, images, sounds and other resource files) can be bundled in a JAR file and subsequently downloaded to a browser in a single HTTP transaction. It also supports file compression and digital signatures.

Java(TM)

is Sun's trademark for a set of technologies for creating and safely running software programs in both stand-alone and networked environments.

Java Application Environment (JAE)

The source code release of the Java Development Kit (JDK(TM)) software.

Java(TM) 2 Platform, Standard Edition (J2SE platform)

The core Java technology platform.

Java(TM) 2 Platform, Enterprise Edition (J2EE platform)

An environment for developing and deploying enterprise applications. The J2EE platform consists of a set of services, application programming interfaces (APIs), and protocols that provide the functionality for developing multi-tiered, Web-based applications.

Java(TM) 2 SDK, Enterprise Edition

Sun's implementation of the J2EE platform. This implementation provides an operational definition of the J2EE platform.

JavaBeans(TM)

A portable, platform-independent reusable component model.

Java Blend(TM)

A product that enables developers to simplify database application development by mapping database records to objects in the Java(TM) programming language (Java objects) and Java objects to databases.

Java Card(TM) API

An ISO 7816-4 compliant application environment focused on smart cards.

JavaCheck(TM)

A tool for checking compliance of applications and applets to a specification.

JavaChip(TM)

Sun's processor, which executes bytecode for the Java(TM) virtual machine***** natively. With a JavaChip processor, bytecode bypasses the virtual machine or just-in-time compiler stage to go directly to the processor.

Java(TM) Compatibility Kit (JCK)

A test suite, a set of tools, and other requirements used to certify a Java platform implementation

conformant both to the applicable Java platform specifications and to Java Software reference implementations.

Java Database Connectivity (JDBC(TM))

An industry standard for database-independent connectivity between the Java(TM) platform and a wide range of databases. The JDBC(TM) provides a call-level API for SQL-based database access.

Java Developer Connection(SM)

A service designed for individual developers, providing online training, product discounts, feature articles, bug information, and early access capabilities.

Java Development Kit (JDK(TM))

A software development environment for writing applets and applications in the Java programming language.

Java(TM) Electronic Commerce Framework

A structured architecture for the development of electronic commerce applications in the Java(TM) programming language.

Java(TM) Enterprise API

This API makes it easy to create large-scale commercial and database applications that can share multimedia data with other applications within an organization or across the Internet. Four APIs have been designed within the Java(TM) Enterprise API family.

Java(TM) Foundation Classes (JFC)

An extension that adds graphical user interface class libraries to the Abstract Windowing Toolkit (AWT).

Java(TM) IDL

A technology that provides CORBA interoperability and connectivity capabilities for the J2EE platform. These capabilities enable J2EE applications to invoke operations on remote network services using the OMG IDL and IIOP.

Java(TM) Interface Definition Language (IDL)

APIs written in the Java programming language that provide standards-based interoperability and connectivity with CORBA (Common Object Request Broker Architecture).

JavaMail(TM)

An API for sending and receiving email.

Java(TM) Media APIs

A set of APIs that support the integration of audio and video clips, 2D fonts, graphics, and images as well as 3D models and telephony.

Java(TM) Media Framework

The core framework supports clocks for synchronizing between different media (e.g., audio and video output). The standard extension framework allows users to do full audio and video streaming.

Java(TM) Message Service (JMS)

An API for using enterprise messaging systems such as IBM MQ Series, TIBCO Rendezvous, and so on.

Java Naming and Directory Interface(TM) (JNDI)

A set of APIs that assists with the interfacing to multiple naming and directory services.

JavaOS(TM)

An Java(TM) technology-based operating system that is optimized to run on a variety of computing and consumer platforms. The JavaOS(TM) operating environment provides a runtime specifically tuned to run applications written in the Java programming language directly on hardware platforms without requiring a host operating system.

JavaPlan(TM)

An object-oriented design and diagramming tool written in the Java(TM) programming language.

Java(TM) Platform

Consists of the Java language for writing programs; a set of APIs, class libraries, and other programs used in developing, compiling, and error-checking programs; and a Java virtual machine which loads and executes the class files.

In addition, the Java platform is subject to a set of compatibility requirements to ensure consistent and compatible implementations. Implementations that meet the compatibility requirements may qualify for Sun's targeted compatibility brands.

Java 2 is the current generation of the Java Platform.

Java(TM) Platform Editions

A Java platform "edition" is a definitive and agreed-upon version of the Java platform that provides the functionality needed over a broad market segment.

An edition is comprised of two kinds of API sets: (i) "core packages," which are essential to all implementations of a given platform edition, and (ii) "optional packages," which are available for a given platform edition and which may be supported in a compatible implementation.

There are 3 distinct editions of the Java Platform:

* Java 2 Platform, Enterprise Edition:
The edition of the Java platform that is targeted at enterprises to enable development, deployment, and management of multi-tier server-centric applications.

* Java 2 Platform, Standard Edition:
The edition of the Java platform that enables development, deployment, and management of cross-platform, general-purpose applications.

* Java 2 Platform, Micro Edition:
The edition of the Java platform that is targeted at small, standalone or connectable consumer and embedded devices to enable development, deployment, and management of applications that can scale from smart cards through mobile devices and set-top boxes to conventional computing devices.

Java(TM) Remote Method Invocation (RMI)

A distributed object model for Java(TM) program to Java program, in which the methods of remote objects written in the Java programming language can be invoked from other Java virtual machines*****, possibly on different hosts.

Java(TM) Runtime Environment (JRE)

A subset of the Java Development Kit (JDK(TM)) for end-users and developers who want to redistribute the runtime environment alone. The Java runtime environment consists of the Java virtual machine*****, the Java core classes, and supporting files.

JavaSafe(TM)

A tool for tracking and managing source file changes, written in the Java(TM) programming language.

JavaScript(TM)

A Web scripting language that is used in both browsers and Web servers. Like all scripting languages, it is used primarily to tie other components together or to accept user input.

JavaServer Pages(TM) (JSP)

An extensible Web technology that uses template data, custom elements, scripting languages, and server-side Java objects to return dynamic content to a client. Typically the template data is HTML or XML elements, and in many cases the client is a Web browser.

JSP action

A JSP element that can act on implicit objects and other server-side objects or can define new scripting variables. Actions follow the XML syntax for elements with a start tag, a body and an end tag; if the body is empty it can also use the empty tag syntax. The tag must use a prefix.

JSP action, standard

An action that is defined in the JSP specification and is always available to a JSP file without being imported.

JSP action, custom

An action described in a portable manner by a tag library descriptor and a collection of Java classes and imported into a JSP page by a taglib directive.

JSP application

A stand-alone Web application written using the JavaServer Pages technology, including JSP files, servlets, HTML files, images, applets, and JavaBeans components.

JSP container

A container that provides the same services as a servlet container and an engine that interprets and processes JSP pages into a servlet.

JSP container, distributed

A JSP container that can run a Web application that is tagged as distributable and that executes across multiple Java virtual machines running on the same host or on different hosts.

JSP declaration

A JSP scripting element that declares methods, variables, or both in a JSP file.

JSP directive

A JSP element that gives an instruction to the JSP container and is interpreted at translation time.

JSP element

A portion of a JSP page that is recognized by a JSP translator. An element can be a directive, an action, or a scripting element.

JSP expression

A scripting element that contains a valid scripting language expression that is evaluated, converted to a String, and placed into the implicit out object.

JSP file

A file named with a .jsp extension that a developer authors using standard HTML tags, core JSP tags, custom JSP tags, and scripting language statements in order to display dynamic pages in a Web browser.

JSP page

A text-based document using fixed template data and JSP elements that describes how to process a request to create a response.

JSP scripting element

A JSP declaration, scriptlet, or expression, whose tag syntax is defined by the JSP specification, and whose content is written according to the scripting language used in the JSP page. The JSP specification describes the syntax and semantics for the case where the language page attribute is "java".

JSP scriptlet

A JSP scripting element containing any code fragment that is valid in the scripting language used in the JSP page. The JSP specification describes what is a valid scriptlet for the case where the language page attribute is "java".

JSP tag

A piece of text between a left angle bracket and a right angle bracket that is used in a JSP file as part of a JSP element. The tag is distinguishable as markup, as opposed to data, because it is surrounded by angle brackets.

JSP tag library

A collection of tags identifying custom actions described via a tag library descriptor and Java classes. A portable Java class library that has a unique URI and that defines custom tags that perform specialized tasks. A JSP tag library can be imported into any JSP file and used with various scripting languages.

Java Studio(TM)

The first program that allows you to easily create Java(TM) technology-based applications and applets without having to know the Java programming language.

Java(TM) Technologies

A set of technologies that enable the creation and safe running of software programs in both stand-alone and networked environments.

Java(TM) Transaction API (JTA)

An API that allows applications and J2EE servers to access transactions.

Java(TM) Transaction Service (JTS)

Specifies the implementation of a transaction manager which supports JTA and implements the Java mapping of the OMG Object Transaction Service (OTS) 1.1 specification at the level below the API.

*Java(TM) virtual machine (JVM)**

A software "execution engine" that safely and compatibly executes the byte codes in Java class files on a microprocessor (whether in a computer or in another electronic device).

Java HotSpot performance engine - the ultra-high-performance engine for a Java runtime environment which features an adaptive compiler that dynamically optimizes the performance of running applications.

KJava virtual machine - the small-footprint, highly optimized foundation of a runtime environment within the Java 2 Platform, Micro Edition. Derived from the Java virtual machine, it is targeted at small connected devices and can scale from 30KB to approximately 128KB, depending on the target device's functionality.

Java Card Virtual Machine - an ultra-small-footprint, highly-optimized foundation of a runtime environmentwithin the Java 2 Platform, Micro Edition. Derived from the Java virtual machine, it is targeted at smart cards and other severely memory-constrained devices and can run in devices with memory as small as 24K of ROM, 16K of EEPROM, and 512 bytes of RAM.

Java Web Server(TM)

The easy-to-use, extensible, easy-to-administer, secure, platform-independent solution to speed and simplify the deployment and management of your Internet and Intranet Web sites. It provides immediate productivity for robust, full-featured, Java technology-based server applications.

Java Workshop(TM)

A complete set of tools integrated into a single environment for managing programming with Java technology. The Java Workshop software uses a highly modular structure that enables you to easily plug new tools into the overall structure.

Java(TM) wallet

A user interface, built on the Java(TM) Electronic Commerce Framework, which allows for online purchases, value transfers, and administrative functions.

JavaSpaces(TM)

A technology that provides distributed persistence and data exchange mechanisms for code in the Java(TM) programming language.

JavaSoft(TM)

A former business unit of Sun Microsystems, Inc., currently known as Sun Microsystems, Inc., Java Software division.

JDBC(TM)

Java(TM) Database Connectivity. An industry standard for database-independent connectivity between the Java platform and a wide range of databases. The JDBC interface provides a call-level API for SQL-based database access.

JDK(TM)

Java(TM) Development Kit software. A software development environment for writing applets and application in the Java programming language.

JFC

Java(TM) Foundation Class. An extension that adds graphical user interface class libraries to the Abstract Windowing Toolkit (AWT).

Jini(TM) Technology

a set of Java APIs that may be incorporated an optional package for any Java 2 Platform Edition. The Jini APIs enable transparent networking of devices and services and eliminates the need for system or network administration intervention by a user.

The Jini technology is currently an optional package available on all Java platform editions.

JMAPI

Java(TM) Management API. A collection of Java programming language classes and interfaces that allow developers to build system, network, and service management applications.

JNDI

Java Naming and Directory Interface(TM). A set of APIs that assist with the interfacing to multiple naming and directory services.

JPEG

Joint Photographic Experts Group. An image file compression standard established by this group. It achieves tremendous compression at the cost of introducing distortions into the image which are almost always imperceptible.

JRE

Java(TM) runtime environment. A subset of the Java Developer Kit for end-users and developers who want to redistribute the runtime environment. The Java runtime environment consists of the Java virtual machine*****, the Java core classes, and supporting files.

Just-in-time (JIT) Compiler

A compiler that converts all of the bytecode into native machine code just as a Java(TM) program is run. This results in run-time speed improvements over code that is interpreted by a Java virtual machine*****.

JVM

Java(TM) Virtual Machine*****. The part of the Java Runtime Environment responsible for interpreting bytecodes.

K

keyword

The Java(TM) programming language sets aside words as keywords - these words are reserved by the language itself and therefore are not available as names for variables or methods.

L

lexical

Pertaining to how the characters in source code are translated into tokens that the compiler can understand.

linker

A module that builds an executable, complete program from component machine code modules. The Java(TM) linker creates a runnable program from compiled classes. See also *compiler*, *interpreter*, *runtime system*.

literal

The basic representation of any integer, floating point, or character value. For example, 3.0 is a double-precision floating point literal, and "a" is a character literal.

local variable

A data item known within a block, but inaccessible to code outside the block. For example, any variable defined within a method is a local variable and can't be used outside the method.

long

A Java(TM) programming language keyword used to define a variable of type long.

M

member

A *field* or *method* of a class. Unless specified otherwise, a member is not static.

method

A function defined in a class. See also *instance method*, *class method*. Unless specified otherwise, a method is not static.

method permission

A permission to invoke a specified group of methods of an enterprise bean's home and remote interfaces.

module

A software unit that consists of one or more J2EE components of the same container type and one deployment descriptor of that type. There are three types of modules: EJB, Web, and application client. Modules can be deployed as stand-alone units or assembled into an application.

Mosaic

A program that provides a simple GUI that enables easy access to the data stored on the Internet. These data may be simple files or hypertext documents. Mosaic was written by a team at *NCSA*.

multithreaded

Describes a program that is designed to have parts of its code execute concurrently. See also *thread*.

mutual authentication

When a client uses a public key certificate to establish its identity and maintains its own security context.

N

native

A Java(TM) programming language keyword that is used in method declarations to specify that the method is not implemented in the same Java source file, but rather in another language.

NCSA

National Center for Supercomputer Applications. See also *Mosaic*.

new

A Java(TM) programming language keyword used to create an instance of a class.

null

The null type has one value, the null reference, represented by the literal null, which is formed from ASCII characters. A null literal is always of the null type.

O

object

The principal building blocks of object-oriented programs. Each object is a programming unit consisting of data (*instance variables*) and functionality (*instance methods*). See also *class*.

object-oriented design

A software design method that models the characteristics of abstract or real objects using classes and objects.

octal

The numbering system using 8 as its base, using the numerals 0-7 as its digits. In programs written in the Java(TM) programming language, octal numbers must be preceded with 0. See also *hexadecimal*.

Optional Packages

The set or sets of APIs in a Java platform edition which are available with and may be supported in a compatible implementation.

Over time, optional packages may become required in an edition as the marketplace requires them.

ORB

Object Request Broker. A library than enables CORBA objects to locate and communicate with one another.

OS principal

A principal native to the operating system on which the J2EE platform is executing.

OTS

Object Transaction Service. A definition of the interfaces that permit CORBA objects to participate in transactions.

overloading

Using one identifier to refer to multiple items in the same scope. In the Java(TM) programming language, you can overload methods but not variables or operators.

overriding

Providing a different implementation of a method in a subclass of the class that originally defined the method.

P

package

A group of *types*. Packages are declared with the `package` keyword.

passivation

The process of transferring an enterprise bean from memory to secondary storage. (See activation.)

peer

In networking, any functional unit in the same layer as another entity.

persistence

The protocol for transferring the state of an entity bean between its instance variables and an underlying database.

PersonalJava(TM)

A Java runtime environment for network-connectable applications on personal consumer devices for home, office, and mobile use.

pixel

The picture element on a display area, such as a monitor screen or printed page. Each pixel is individually accessible.

POA

Portable Object Adapter. A CORBA standard for building server-side applications that are portable across heterogeneous ORBs.

POSIX

Portable Operating System for UNIX(TM). A standard that defines the language interface between the UNIX operating system and application programs through a minimal set of supported functions.

primary key

An object that uniquely identifies an entity bean within a home.

principal

The identity assigned to an entity as a result of authentication.

private

A Java(TM) programming language keyword used in a method or variable declaration. It signifies that the method or variable can only be accessed by other elements of its class.

privilege

A security attribute that does not have the property of uniqueness and which may be shared by many principals. An example of a privilege is a group.

process

A virtual address space containing one or more threads.

property

Characteristics of an object that users can set, such as the color of a window.

Profiles

A Profile is a collection of Java APIs that complements one or more Java 2 Platform Editions by adding domain-specific capabilities. Profiles may also include other defined Profiles. A profile implementation requires a Java 2 Platform Edition to create a complete development and deployment environment in a targeted vertical market. Each profile is subject to an associated set of compatibility requirements.

Profiles may be usable on one or more editions.

Some examples of profiles within the Java 2 Platform, Micro Edition are:

* PersonalJava(TM) - for non-PC products that need to display web-compatible Java-based content

* Java Card(TM) - for secure smart cards and other severely memory-constrained devices.

protected

A Java(TM) programming language keyword used in a method or variable declaration. It signifies that the method or variable can only be accessed by elements residing in its class, subclasses, or classes in the same package.

public

A Java(TM) programming language keyword used in a method or variable declaration. It signifies that the method or variable can be accessed by elements residing in other classes.

Q 

R 

raster

A two-dimensional rectangular grid of pixels.

realm

See security policy domain. Also, a string, passed as part of an HTTP request during basic authentication, that defines a protection space. The protected resources on a server can be partitioned into a set of protection spaces, each with its own authentication scheme and/or authorization database.

re-entrant enterprise bean

An enterprise bean that can handle multiple simultaneous, interleaved, or nested invocations which will not interfere with each other.

reference

A data element whose value is an address.

Reference Implementation

See Java 2 SDK, Enterprise Edition.

remote interface

One of two interfaces for an enterprise bean. The remote interface defines the business methods callable by a client.

remove method

Method defined in the home interface and invoked by a client to destroy an enterprise bean.

resource adapter

A system-level software driver that is used by an EJB container or an application client to connect to an EIS. A resource adapter is typically specific to an EIS. It is available as a library and is used within the address space of the server or client using it. A resource adapter plugs in to a container. The application components deployed on the container then use the client API (exposed by adapter) or tool generated high-level abstractions to access the underlying EIS. The resource adapter and EJB container collaborate to provide the underlying mechanisms - transactions, security, and connection pooling - for connectivity to the EIS.

resource manager

Provides access to a set of shared resources. A resource manager participates in transactions that are externally controlled and coordinated by a transaction manager. A resource manager is typically in different address space or on a different machine from the clients that access it. Note: An EIS is referred to as resource manager when it is mentioned in the context of resource and transaction management.

resource manager connection

An object that represents a session with a resource manager.

resource manager connection factory

An object used for creating a resource manager connection.

return

A Java(TM) programming language keyword used to finish the execution of a method. It can be followed by a value required by the method definition.

RFE

Request for Enhancement.

RMI

See *Java Remote Method Invocation*.

role (development)

The function performed by a person in the development and deployment phases of an application developed using J2EE technology. The roles are: Application Application Component Provider, Application Assembler, Deployer, J2EE Platform Provider, EJB Container Provider, EJB Server Provider, Web Container Provider, Web Server Provider, Tool Provider, and System Administrator.

role (security)

An abstract logical grouping of users that is defined by the Application Assembler. When an application is deployed, the roles are mapped to security identities, such as principals or groups, in the operational environment.

role mapping

The process of associating the groups and/or principals recognized by the container to security roles specified in the deployment descriptor. Security roles have to be mapped by the Deployer before the component is installed in the server.

rollback

The point in a transaction when all updates to any databases involved in the transaction are reversed.

root

In a hierarchy of items, the one item from which all other items are descended. The root item has nothing above it in the hierarchy. See also *hierarchy*, *class*, *package*.

RPC

Remote Procedure Call. Executing what looks like a normal procedure call (or method invocation) by sending network packets to some remote host.

runtime system

The software environment in which programs compiled for the Java(TM) virtual machine* can run. The runtime system includes all the code necessary to load programs written in the Java programming language, dynamically link native methods, manage memory, handle exceptions, and an implementation of the Java virtual machine, which may be a Java interpreter.

S

SAX

Simple API for XML. An event-driven, serial-access mechanism for accessing XML documents.

Sandbox

Comprises a number of cooperating system components, ranging from security managers that execute as part of the application, to security measures designed into the Java(TM) virtual machine* and the language itself. The sandbox ensures that an untrusted, and possibly malicious, application cannot gain access to system resources.

scope

A characteristic of an identifier that determines where the identifier can be used. Most identifiers in the Java(TM) programming environment have either class or local scope. Instance and class variables and methods have class scope; they can be used outside the class and its subclasses only by prefixing them with an instance of the class or (for class variables and methods) with the class name. All other variables are declared within methods and have local scope; they can be used only within the enclosing block.

Secure Socket Layer (SSL)

A protocol that allows communication between a Web browser and a server to be encrypted for privacy.

security attributes

A set of properties associated with a principal. Security attributes can be associated with a principal by an authentication protocol and/or by a J2EE Product Provider.

security constraint

A declarative way to annotate the intended protection of Web content. A security constraint consists of a Web resource collection, an authorization constraint, and a user data constraint.

security context

An object that encapsulates the shared state information regarding security between two entities.

security permission

A mechanism, defined by J2SE, used by the J2EE platform to express the programming restrictions imposed on Application Component Providers.

security permission set

The minimum set of security permissions that a J2EE Product Provider must provide for the execution of each component type.

security policy domain

A scope over which security policies are defined and enforced by a security administrator. A security policy domain has the following characteristics:

It has a collection of users (or principals).
It uses a well defined authentication protocol(s) for authenticating users (or principals).
It may have groups to simplify setting of security policies.

security role

See role (security).

security technology domain

A scope over which the same security mechanism is used to enforce a security policy. Multiple security policy domains can exist within a single technology domain.

server principal

The OS principal that the server is executing as.

servlet

A Java program that extends the functionality of a Web server, generating dynamic content and interacting with Web clients using a request-response paradigm.

servlet container

A container that provides the network services over which requests and responses are sent, decodes requests, and formats responses. All servlet containers must support HTTP as a protocol for requests and responses, but may also support additional request-response protocols such as HTTPS.

servlet container, distributed

A servlet container that can run a Web application that is tagged as distributable and that executes across multiple Java virtual machines running on the same host or on different hosts.

servlet context

An object that contains a servlet's view of the Web application within which the servlet is running. Using the context, a servlet can log events, obtain URL references to resources, and set and store attributes that other servlets in the context can use.

servlet mapping

Defines an association between a URL pattern and a servlet. The mapping is used to map requests to servlets. If the container handling the request is a JSP container, a URL containing a .jsp extension is implicitly mapped.

session

An object used by a servlet to track a user's interaction with a Web application across multiple HTTP requests.

session bean

An enterprise bean that is created by a client and that usually exists only for the duration of a single client/server session. A session bean performs operations, such as calculations or accessing a database, for the client. While a session bean may be transactional, it is not recoverable should a system crash occur. Session bean objects can be either stateless or they can maintain conversational state across methods and transactions. If they do maintain state, then the EJB container manages this state if the object must be removed from memory. However, the session bean object itself must manage its own persistent data.

short

A Java(TM) programming language keyword used to define a variable of type short.

single precision

In the Java(TM) language specification, describes a floating point number with 32 bits of data. See also *double precision*.

SGML

Standardized Generalized Markup Language. An ISO/ANSI/ECMA standard that specifies a way to annotate text documents with information about types of sections of a document.

SQL

Structured Query Language. The standardized relational database language for defining database objects and manipulating data.

stateful session bean

A session bean with a conversational state.

stateless session bean

A session bean with no conversational state. All instances of a stateless session bean are identical.

System Administrator

The person responsible for configuring and administering the enterprise's computers, networks, and software systems.

static

A Java(TM) programming language keyword used to define a variable as a class variable. Classes maintain one copy of class variables regardless of how many instances exist of that class. "static" can also be used to define a method as a class method. Class methods are invoked by the class instead of a specific instance, and can only operate on class variables.

static field

Another name for *class variable*.

static method

Another name for *class method*.

subarray

An array that is inside another array.

subclass

A class that is derived from a particular class, perhaps with one or more classes in between. See also *superclass*, *supertype*.

subtype

If type X *extends* or implements type Y, then X is a subtype of Y. See also *supertype*.

superclass

A class from which a particular class is derived, perhaps with one or more classes in between. See also *subclass*, *subtype*.

super

A Java(TM) programming language keyword used to access members of a class inherited by the class in which it appears.

supertype

The supertypes of a type are all the interfaces and classes that are extended or implemented by that type. See also *subtype*, *superclass*.

switch

A Java(TM) programming language keyword used to evaluate a variable that can later be matched with a value specified by the "case" keyword in order to execute a group of statements.

Swing Set

The code name for a collection of graphical user interface (GUI) components that runs uniformly on any native platform which supports the Java(TM) virtual machine*****. Because they are written entirely in the Java programming language, these components may provide functionality above and beyond that provided by native-platform equivalents. (Contrast with AWT.)

synchronized

A keyword in the Java programming language that, when applied to a method or code block, guarantees that at most one thread at a time executes that code.

T

TCP/IP

Transmission Control Protocol based on IP. This is an Internet protocol that provides for the reliable delivery of streams of data from one host to another. See also *IP*.

Technology Compatibility Kit (TCK)

A test suite, a set of tools, and other requirements used to certify an implementation of a particular Sun technology conformant both to the applicable specifications and to Sun or Sun-designated reference implementations.

Thin Client

A system that runs a very light operating system with no local system administration and executes applications delivered over the network.

this

A Java(TM) programming language keyword that can be used to represent an instance of the class in which it appears. "this" can be used to access class variables and methods.

thread

The basic unit of program execution. A process can have several threads running concurrently, each performing a different job, such as waiting for events or performing a time-consuming job that the program doesn't need to complete before going on. When a thread has finished its job, the thread is suspended or destroyed. See also *process*.

throw

A Java(TM) programming language keyword that allows the user to throw an exception or any class that implements the "throwable" interface.

throws

A Java(TM) programming language keyword used in method declarations that specify which exceptions are not handled within the method but rather passed to the next higher level of the program.

Tool Provider

An organization or software vendor that provides tools used for the development, packaging, and deployment of J2EE applications.

transaction

An atomic unit of work that modifies data. A transaction encloses one or more program statements, all of which either complete or roll back. Transactions enable multiple users to access the same data concurrently.

transaction attribute

A value specified in an enterprise bean's deployment descriptor that is used by the EJB container to control the transaction scope when the enterprise bean's methods are invoked. A transaction attribute can have the following values:
Required, RequiresNew, Supports, NotSupported, Mandatory, Never.

transaction isolation level

The degree to which the intermediate state of the data being modified by a transaction is visible to other concurrent transactions and data being modified by other transactions is visible to it.

transaction manager

Provides the services and management functions required to support transaction demarcation, transactional resource management, synchronization, and transaction context propagation.

transient

A keyword in the Java programming language that indicates that a field is not part of the serialized form of an object. When an object is serialized, the values of its transient fields are not included in the serial representation, while the values of its non-transient fields are included.

try

A Java(TM) programming language keyword that defines a block of statements that may throw a Java language exception. If an exception is thrown, an optional "catch" block can handle specific exceptions thrown within the "try" block. Also, an optional "finally" block will be executed regardless of whether an exception is thrown or not.

type

A class or interface.

U

Unicode

A 16-bit character set defined by ISO 10646. See also *ASCII*. All source code in the Java(TM) programming environment is written in Unicode.

URI

Uniform Resource Identifier. A compact string of characters for identifying an abstract or physical resource. A URI is either a URL or a URN. URLs and URNs are concrete entities that actually exist; A URI is an abstract superclass.

URL

Uniform Resource Locator. A standard for writing a text reference to an arbitrary piece of data in the WWW. A URL looks like "protocol://host/localinfo" where protocol specifies a protocol to use to fetch the object (like HTTP or FTP), host specifies the Internet name of the host on which to find it, and localinfo is a string (often a file name) passed to the protocol handler on the remote host.

URL path

The URL passed by a HTTP request to invoke a servlet. The URL consists of the Context Path + Servlet Path + PathInfo, where Context Path is the path prefix associated with a servlet context that this servlet is a part of. If this context is the default context rooted at the base of the web server's URL namespace, the path prefix will be an empty string. Otherwise, the path prefix starts with a / character but does not end with a / character. Servlet Path is the path section that directly corresponds to the mapping which activated this request. This path starts with a / character. PathInfo is the part of the request path that is not part of the Context Path or the Servlet Path.

URN

Uniform Resource Name. A unique identifier that identifies an entity, but doesn't tell where it is located. A system can use a URN to look up an entity locally before trying to find it on the Web. It also allows the Web location to change, while still allowing the entity to be found.

user data constraint

Indicates how data between a client and a container should be protected. The protection can be the prevention of tampering with the data or prevention of eavesdropping on the data.

V

variable

An item of data named by an identifier. Each variable has a type, such as `int` or `Object`, and a

scope. See also *class variable*, *instance variable*, *local variable*.

virtual machine

An abstract specification for a computing device that can be implemented in different ways, in software or hardware. You compile to the instruction set of a virtual machine much like you'd compile to the instruction set of a microprocessor. The Java(TM) virtual machine* consists of a bytecode instruction set, a set of registers, a stack, a garbage-collected heap, and an area for storing methods.

vocabulary

Traditionally, software programs have been written and then compiled into machine code that is directly dependent on the operating system that drives the microprocessor in the computer. The Java platform mitigates this dependency by providing a model in which software is written and compiled, and can then be transmitted over a network and run anywhere a fully compliant virtual machine is present.

This model provides the additional benefit of heightened security, both because programs can be verified by the virtual machine after they have been transmitted over a network, and because the virtual machine can run programs in a secure "sandbox" that prevents certain destructive behaviors.

Software programmers have embraced the Java platform because it reduces the cost and time required to write and support software code. They are no longer required to rewrite software to function on different computers with different operating systems and microprocessors. Companies and organizations deploying applications favor Java technology because it minimizes the cost of purchasing or modifying different versions of software applications for the various types of computers and servers within their networks.

void

A Java(TM) programming language keyword used in method declarations to specify that the method does not return any value. "void" can also be used as a nonfunctional statement.

volatile

A Java(TM) programming language keyword used in variable declarations that specifies that the variable is modified asynchronously by concurrently running threads.

W

wait

A UNIX® command which will wait for all background processes to complete, and report their termination status.

Web application, distributable

A Web application that uses J2EE technology written so that it can be deployed in a Web container distributed across multiple Java virtual machines running on the same host or different hosts. The deployment descriptor for such an application uses the distributable element.

Web component

A component that provides services in response to requests; either a servlet or a JSP page.

Web container

A container that implements the Web component contract of the J2EE architecture. This contract specifies a runtime environment for Web components that includes security, concurrency, life cycle management, transaction, deployment, and other services. A container that provides the same services as a JSP container and federated view of the J2EE platform APIs. A Web container is provided by a Web or J2EE server.

Web container, distributed

A Web container that can run a Web application that is tagged as distributable and that executes across multiple Java virtual machines running on the same host or on different hosts.

Web server

Software that provides services to access the Internet, an intranet, or an extranet. A Web server hosts Web sites, provides support for HTTP and other protocols, and executes server-side programs (such as CGI scripts or servlets) that perform certain functions. In the J2EE architecture, a Web server provides services to a Web container. For example, a Web container typically relies on a Web server to provide HTTP message handling. The J2EE architecture assumes that a Web container is hosted by a Web server from the same vendor, so does not specify the contract between these two entities. A Web server may host one or more Web containers.

while

A Java(TM) programming language keyword used to declare a loop that iterates a block of statements. The loop`s exit condition is specified as part of the while statement.

world readable files

Files on a file system that can be viewed (read) by any user. For example: files residing on web servers can only be viewed by Internet users if their permissions have been set to world readable.

wrapper

An object that encapsulates and delegates to another object to alter its interface or behavior in some way.

WWW

World Wide Web. The web of systems and the data in them that is the Internet. See also *Internet*.

X

XML

Extensible Markup Language. A markup language that allows you to define the tags (markup) needed to identify the data and text in XML documents. J2EE deployment descriptors are expressed in XML.

Y

Z

***As used on this web site, the terms "Java virtual machine" or "JVM" mean a virtual machine for**

the Java platform.